

Publication number: JP2001044848

Publication date: 2001-02-16

Inventor: NAKAYAMA TADAYOSHI

Applicant: CANON KK

Classification:

- international: H04N1/41; H03M7/40; H04N1/413; H04N7/24;
H04N7/26; H04N1/41; H03M7/40; H04N1/413;
H04N7/24; H04N7/26; (IPC-1-7): H03M7/40; H04N1/41;
H04N1/413; H04N7/24

- European:

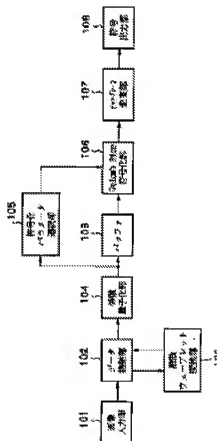
Application number: JP19990212713 19990727

Priority number(s): JP19990212713 19990727

[Report a data error here](#)

Abstract of JP2001044848

PROBLEM TO BE SOLVED: To simply apply variable length coding to a multi-valued image at a high speed. **SOLUTION:** A data storage section 102 stores multi-valued image data received from an image input section 101, and a discrete wavelet conversion section 103 applies conversion processing to the stored data. As a result, data of sub blocks by each frequency component are fed to a coefficient quantization section 104, quantized according to quantization coefficient set by each sub block, and the quantized data are stored in a buffer 108. A coding parameter selection section 105 decides the optimum shift parameter for each quantized sub block and notifies a Golomb Rice coding section 160 of this.



Data supplied from the **esp@cenet** database - Worldwide

Family list5 family members for: **JP2001044848**

Derived from 5 applications

[Back to JP2001044](#)

- 1 **CODER, METHOD THEREFOR AND STORAGE MEDIUM**
Inventor: NAKAYAMA TADAYOSHI Applicant: CANON KK
EC: IPC: **H04N1/41; H03M7/40; H04N1/413** (+11)
Publication info: **JP2001044848 A** - 2001-02-16
- 2 **ENCODER**
Inventor: NAKAYAMA TADAYOSHI Applicant: CANON KK
EC: IPC: **H04N7/26; H03M7/40; H04N1/41** (+8)
Publication info: **JP2001077699 A** - 2001-03-23
- 3 **ENCODER AND ENCODING METHOD**
Inventor: NAKAYAMA TADAYOSHI Applicant: CANON KK
EC: IPC: **G06F5/00; H03M7/30; H03M7/40** (+12)
Publication info: **JP2001298623 A** - 2001-10-26
- 4 **CODER AND ITS METHOD**
Inventor: NAKAYAMA TADAYOSHI Applicant: CANON KK
EC: IPC: **H04N7/30; H03M7/30; H03M7/40** (+8)
Publication info: **JP2001298739 A** - 2001-10-26
- 5 **Coding apparatus and method**
Inventor: NAKAYAMA TADAYOSHI (JP) Applicant: CANON KK (JP)
EC: H04N7/26L; H04N7/26A4V; (+4) IPC: **G06K9/46; H04N7/26; G06K9/46** (+2)
Publication info: **US6865299 B1** - 2005-03-08

Data supplied from the **esp@cenet** database - Worldwide

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2001-44848

(P2001-44848A)

(43) 公開日 平成13年2月16日 (2001.2.16)

(51) Int.Cl. ⁷	識別記号	F I	ページコード ⁸ (参考)
H 0 3 M 7/40		H 0 3 M 7/40	5 C 0 5 9
H 0 4 N 1/41		H 0 4 N 1/41	B 5 C 0 7 8
1/413		1/413	Z 5 J 0 6 4
7/24		7/13	Z 9 A 0 0 1

審査請求 未請求 請求項の数15 O L (全 17 頁)

(21) 出願番号 特願平11-212713

(22) 出願日 平成11年7月27日 (1999.7.27)

(71) 出願人 000001007

キヤノン株式会社

東京都大田区下丸子3丁目30番2号

(72) 発明者 中山 忠義

東京都大田区下丸子3丁目30番2号 キヤ

ノン株式会社内

(74) 代理人 100076428

弁理士 大塚 康徳 (外2名)

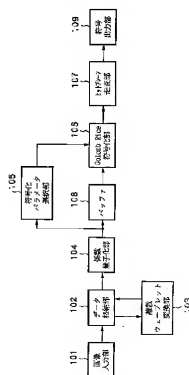
最終頁に続く

(54) 【発明の名称】 符号化装置及び方法及び記憶媒体

(57) 【要約】

【課題】 多値画像を簡易かつ高速に可変長符号化することのできるようになる。

【解決手段】 画像入力部101より入力した多値画像データはデータ格納部102に格納され、離散ウェーブレット変換部103により変換処理が施される。この結果、各周波数成分毎のサブブロックのデータが係数量子化部104に供給され、各サブブロック毎に設定された量子化係数に従って量子化され、バッファ105に格納される。符号化パラメータ選択部105は、量子化された各サブブロック毎に、最適なシフトパラメータを決定し、それをGolomb-Rice符号化部106に通知する。



【特許請求の範囲】

【請求項1】 あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を評価することにより、前記パラメータの最適値を決定することを特徴とする符号化方法。

【請求項2】 前記可変長符号部分の符号量の差分値は、パラメータの値が1だけ異なる場合、あるいは、2以上異なる場合の差分値であることを特徴とする前記請求項1記載の符号化方法。

【請求項3】 前記可変長符号部分の符号量の差分値を計算する時に、符号化するシンボル群の各ビットプレーン中の1の個数を表わす情報を用いることを特徴とする前記請求項1記載の符号化方法。

【請求項4】 前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列を、端から評価することを特徴とする前記請求項1記載の符号化方法。

【請求項5】 前記可変長符号部分の符号量の差分値を、パラメータ値の昇順あるいは降順に並べた差分値列の途中の値を評価した結果に基づき、それ以降の評価順序を決定することを特徴とする前記請求項1記載の符号化方法。

【請求項6】 あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量と、該符号量に所定の値を加算した値とを比較結果に基づいて、前記パラメータの最適値を決定することを特徴とする符号化方法。

【請求項7】 あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化装置において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を計算する手段と、該差分値を所定値と比較する手段と、該比較結果に基づいて前記パラメータの最適値を決定する手段とを有することを特徴とする符号化装置。

【請求項8】 前記可変長符号部分の符号量の差分値は、パラメータの値が1だけ異なる場合、あるいは、2以上異なる場合の差分値であることを特徴とする請求項7項記載の符号化装置。

【請求項9】 前記可変長符号部分の符号量の差分値を計算する手段は、符号化するシンボル群の各ビットプレーン中の1の個数を計数する手段を併せ持つことを特徴とする請求項7項記載の符号化装置。

【請求項10】 前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列を、端から所定値と順次比較する手段を有することを特

徴とする請求項7項記載の符号化装置。

【請求項11】 前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列の途中の値を所定値と比較する手段と、該比較結果に基づき、それ以降の比較順序を制御する手段と、該制御手段のもとで比較を行なう手段とを有することを特徴とする請求項7項記載の符号化装置。

【請求項12】 あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量を計算する手段と、該符号量に所定の値を加算する手段と、前記符号量と前記加算結果とを比較する手段と、該比較結果に基づいて、前記パラメータの最適値を決定する手段とを有することを特徴とする符号化装置。

【請求項13】 二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化装置であって、前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化する手段と、量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定する手段とを備えることを特徴とする符号化装置。

【請求項14】 二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化装置であって、前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化し、量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定することを特徴とする符号化方法。

【請求項15】 コンピュータが読み込み実行することで、二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化装置として機能するプログラムコードを格納する記憶媒体であって、前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化し、量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定するプログラムコードを格納する記憶媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は符号化装置及び方法

及び記憶媒体に関するものである。

【0002】

【従来の技術】画像、特に多値画像は非常に多くの情報を含んでおり、その画像を蓄積・伝送する際にはデータ量が膨大になってしまうという問題がある。このため画像の蓄積・伝送に際しては、画像の持つ冗長性を取り除き、画質の劣化を多少許すことで、データ量の削減を図る高効率符号化が用いられている。

【0003】これら高効率符号化の1つとしてJPEG方式の圧縮符号化が知られている。この方式では多値画像をブロック毎にDCT変換することにより周波数成分に変換し、得られた変換係数を量子化し、可変長符号化するというものである。

【0004】上記DCT変換を用いた符号化は、圧縮率を高く設定すると復号画像にブロック歪みが発生してしまうという問題がある。近年、このような歪みを解消するためにウェーブレット変換を用いる新たな符号化方式が提案されている。

【0005】また、これら種々の高効率符号化の一部として、可変長符号化が適用されることが多い。

【0006】

【発明が解決しようとする課題】しかしながら、多値画像を簡単な処理で可変長符号化する為の方式、更にはこの可変長符号化方式を高速に実行するための処理方法については未だ確立されていない。

【0007】本発明はかかる状況に鑑みてなされたものであり、多値画像を簡易かつ高速に可変長符号化することのできる符号化装置及び方法及び記憶媒体を提供することを目的とする。

【0008】

【課題を解決するための手段】この課題を解決するため、例えば本発明の符号化装置は以下の構成を備える。すなわち、あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を評価することにより、前記パラメータの最適値を決定する。

【0009】また、本発明の好適な実施態様に従えば、多値画像データを複数の情報群（本実施の形態では1サブブロック分の量子化値の集まり）に分割し、該情報群を各々個別のパラメータ（kパラメータ）に基づき可変長符号化（Golomb Rice符号化）する為の符号化装置であって、可変長符号部分の符号量が隣接するkパラメータ間でどのくらい差があるのか、その差分値を求め、kパラメータの許容範囲内で求めた全ての差分値を、所定の値と順次比較することにより、各情報群に最適なパラメータを決定する。

【0010】

【発明の実施の形態】以下、添付図面に従って本発明に

係る実施形態を詳細に説明する。

【0011】[第1の実施形態]以下、8ビットのモノクロ画像データを符号化する例を第1の実施形態として説明するが、本発明はこれに限らず、各画素4ビットで表わすモノクロ画像、或いは各画素における色成分（RGB/Lab/YCrCb）を8ビットで表現するカラーの多値画像を符号化する場合に適用することも可能である。これらに適用する場合には、各画素成分をそれぞれモノクロ画像として符号化すればよい。

【0012】図1は第1の実施形態における画像符号化装置のブロック構成図である。

【0013】同図において、101は画像データ入力部、102はデータ格納部、103は離散ウェーブレット変換部、104は係数量子化部、105は符号化パラメータ選択部、106はGolomb Rice符号化部、107はビットアレーン走査部、108はバッファ、109は符号出力部である。

【0014】まず、画像入力部101では符号化対象となる画素データをラスタースキャン順に入力する。この画像入力部101は、例えばスキャナ、デジタルカメラ等の撮像装置、或いはCCDなどの撮像デバイス、或いはネットワーク回線のインターフェース等からの画像データ入力処理があげられる。

【0015】また、画像入力対象はRAM、ROM、ハードディスク、CD-ROM等の記録媒体であってもよい。

【0016】画像データ格納部102は画像入力部101で入力される画像データを、記憶する。

【0017】離散ウェーブレット変換部103では、上記画像データ格納部に記憶した1画面分の画像データに対して公知の離散ウェーブレット変換を施し、複数の周波数帯域に分解する。本実施形態の形態では、画像データ列 $x(n)$ に対する離散ウェーブレット変換は次式によって行なうものとする。

$$【0018】r(n) = floor\{(x(2n) + x(2n+1))/2\}$$

$$d(n) = x(2n+2) - x(2n+3) + floor\{(-r(n) + r(n+2) + 2)/4\}$$

$r(n)$ 、 $d(n)$ は変換係数であり、 $r(n)$ は低周波成分、 $d(n)$ は高周波成分である。

【0019】また上式において $floor\{X\}$ は X を超えない最大の整数値を表わす。本変換式は一次元のデータに対するものであるが、この変換を水平方向、垂直方向の順に適用することで二次元の変換を行なうことが可能であり、図2(a)の様なLL、HL、LH、HHの4つの周波数帯域（サブブロック）に分割することができる。これらの変換データは、次段の係数量子化部104へスムーズにデータを出力したり、あるいは、更なるウェーブレット変換を行なうため、前記データ格納部102に記憶する。

【0020】生成したL/L成分について前記同様の手順にて離散ウェーブレット変換を施すことにより図2

(b)のように7個の周波数帯域(サブブロック)に分解する。本実施形態の形態においては、更にもう一度繰り返して離散ウェーブレット変換を施すことにより図2(c)に示すようにL/L, HL3, LH3, HH3, HL2, LH2, HH2, HL1, LH1, HH1の10個の周波数帯域(サブブロック)に分割し、前記データ格納部102に記憶する。

【0021】変換係数はL/L, HL3, LH3, HH3, HL2, LH2, HH2, HL1, LH1, HH1のサブブロック順に、かつ各サブブロック毎にラスタースキャン順にデータ格納部102から係数量子化部104へと出力する。

【0022】係数量子化部104はウェーブレット変換係数の各々を各周波数成分毎に定めた量子化ステップで量子化し、量子化後の値を符号化パラメータ選択部105及びバッファ108へと出力する。

【0023】係数値をX、この係数の属する周波数成分に対する量子化ステップの値をqとすると、量子化後の量子化値Q(X)は次式によって求めるものとする。

$$Q(X) = \text{floor} \{ (X/q) + 0.5 \}$$

本実施形態の形態における各周波数成分と量子化ステップとの関係を図3に示す。図に示すように低周波成分(L/L等)よりも高周波成分(HL1, LH1, HH1)等の方が量子化ステップを大きくしている。これにより、視覚的に劣化の目立ちにくい高周波成分の情報の削減を行ない圧縮の向上を図っている。

【0025】上記量子化値は、符号化パラメータ選択部105で評価され、後段のGolomb Rice符号化部106でこの量子化値を符号化する際に必要となるkパラメータが決定されるまで、バッファ108に蓄積保持される。

【0026】符号化パラメータ選択部105では、上記量子化値に基づいて後段のGolomb Rice符号化部106で使用されるkパラメータを選択する。

【0027】このkパラメータとはGolomb Rice符号化という可変長符号化を行なう際の固定長部の符号長を示す値である。説明の都合上、符号化パラメータ選択部105の説明する前に、Golomb Rice符号化部106の動作から先に説明する。

【0028】このGolomb Rice符号化部106で行なう符号化の基本的な方法は公知ではあるが、以下に符号化の基本的な動作及び本発明の特徴的な部分について説明する。

【0029】まず、量子化値を格納したバッファ108から1サブブロック分のデータを順次読み出す。

【0030】次に、Golomb Rice符号化部106で、該サブブロックに対応するkパラメータに基づ

き、以下に説明する前処理やGolomb Rice符号化処理を行なう。

【0031】Golomb Rice符号化部106は入力した各量子化値の正/負を調べ、負号ビットを出力する。具体的には量子化値が0または正である場合には「0」を、負である場合には「1」を出力した後、該量子化値を絶対値に変換する。(この処理は既述にはGolomb Rice符号化処理には含まれないが、ここでは前処理として、Golomb Rice符号化部106内で行なうものとする)

次に、該絶対値をGolomb Rice符号化する。符号化対象となる量子化値の絶対値をV、処理対象のサブブロックに適用されるkパラメータの値がkである場合のGolomb Rice符号化処理は次の手順にて行なう。

【0032】まず、2進表現したVをkビット右シフトした後に、得られた値(残った値)の個数だけ「0」を連続して配置し、その次に区切りのビットとして「1」を配置し、その次に元のVの下位kビットを配置することにより可変長符号(Golomb Rice符号)を生成する。

【0033】図4にV=0~7、k=0, 1, 2に対するGolomb Rice符号の例を示す。例えば、V=4で、k=1の場合、4は二進数で「100B(Bは二進数を示す)」であるから、k(=1)ビット右シフトすると、残りは「10B」、つまり「2」であるから、「0」を2回続け、その後区切りの「1」、そして値4(=100B)の下位k(=1)ビットにより、「0010」を得る。

【0034】これから分かるように、絶対値V及びパラメータkから得られる各可変長符号の符号長は、 $V > k$ (Vをkビット右シフトしたときに残った値)+1 (区切りビット)+k (パラメータの値) ビットであることが容易に推測できる。正/負を表わす負号ビットを加えればさらに1ビット増える。

【0035】また、図4からも分かるように、kパラメータ=0の時には量子化値「0」に対するGolomb Rice符号長がk=1, 2の時よりも特に短くなっている。これは符号化される量子化値群が0に偏っているため、kパラメータを小さくして符号化の方が適していることを示している。

【0036】また、ここで用いるGolomb Rice符号化は、図4に示すような符号表を実際には保持することなく簡単な演算により符号化及び復号化を行なうことができるという効果がある。通常、ハフマン符号化等の可変長符号化では、符号化対象値に対する可変長符号を示すテーブルを保持しなければならぬ。特に、関連する状態に応じ、符号化対象値に対して複数の可変長符号を適宜切り換える場合には、上記テーブルを複数持つ必要がある。

【0037】以上のようにして、入力される量子化値に対する負号ビット（+/-を表わす）と可変長符号（Golomb-Rice符号）からなる符号化データを生成し、後段のビットプレーン走査部107へ出力する。

【0038】次に、上述したGolomb-Rice符

$$T_p = \Sigma (Vi > p \text{ (} Vi \text{を} p \text{ビット右シフトした値)} + 2 \text{ (負号ビットと区切りビット)} + p) \cdots (1)$$

となる。

【0040】符号化するサブブロック内の量子化値の数をN個とすると、上記(1)式は下式ようになる。

【0041】

$$T_p = (p+2) \times N + \Sigma Vi > p \cdots (2)$$

上記(2)式の第1項：(p+2)×Nは固定長符号部分による符号量で、第2項：ΣVi > pは可変長符号部分による符号量である。ここで第2項のみを

$$S_p = \Sigma Vi > p \cdots (3)$$

と表わす。このS_pはpの値が小さい程大きくなる。一方、固定長符号部分の符号量はpの値に比例して大きくなり、pが1つ大きくなると該符号量はNビット増える。

【0042】次に、Viを2進表現したときの各ビットプレーンにおける「1」の個数について考える。

【0043】Vi > pの最下位ビットのプレーン中に存在する「1」の数をB_{p+1}とする。すなわち、
B_{p+1} = Σ (Vi > p) & 1 ⋯ (4)

である。ここで、B₁はViの最下位ビットプレーン中の「1」の個数を表わし、B₂はViの最下位ビットから2番目のプレーン中の「1」の個数を表わす。

$$\text{【0044】} \Sigma Vi > p = (2 \times \Sigma Vi > (p+1)) + (\Sigma (Vi > p) \& 1)$$

という関係と、(3)(4)式から

$$S_p = 2 \times S_{p+1} + B_{p+1} \cdots (5)$$

という関係式が得られる。kパラメータの値が1つだけ異なるS_{p-1}とS_p間の差をD_pとすると、

$$\begin{aligned} D_p &= S_{p-1} - S_p = (2 \times S_p + B_p) - S_p \\ &= S_p + B_p \text{ (} p \geq 1 \text{)} \cdots (6) \end{aligned}$$

このD_pはkパラメータの値がpからp-1に変わることで、可変長符号量がどれだけ増えるかを表わすもので、この値がNより小さいなら、全体の符号量はN-D_pだけ減るので、kパラメータの値はよりp-1にした方がよい。逆に、D_p ≥ Nならkパラメータの値はpの方がよい。

【0045】同様に、kパラメータの値が2つ異なるS_{p-2}とS_p間の差をDD_p、すなわち、DD_p = S_{p-2} - S_pとすると、この値が2Nより小さいなら、kパラメータの値がpからp-2に変わることで可変長符号量が2N-DD_pだけ減るので、kパラメータの値はよりp-2にした方がよい。逆に、DD_p ≥ 2Nならkパラメータの値はpの方がよいと言える。

【0046】上記(6)式をさらに変形すると、

号化部106で必要となるkパラメータを生成する符号化パラメータ選択部150の処理内容の説明を行なう。

【0039】符号化するサブブロックの全符号量Tは、kパラメータの値pとすると

$$\begin{aligned} D_{p-1} &= S_{p-1} + B_{p-1} \\ &= (2 \times S_p + B_p) + B_{p-1} \\ &= (S_p + B_p) + S_p + B_{p-1} \\ &= D_p + S_p + B_{p-1} \\ S_p, B_{p-1} &\geq 0 \text{ であるから} \\ D_{p-1} &\geq D_p \cdots (7) \end{aligned}$$

上記(7)式の関係と上述した説明から次のことが言える。

【0047】『数列：D₁、D₂、D₃、⋯、D_{p-1}、D_p、D_{p+1}、⋯、D_qには、D₁ ≥ D₂ ≥ D₃ ≥ ⋯ ≥ D_{p-1} ≥ D_p ≥ D_{p+1} ≥ ⋯ ≥ D_qという関係があり、D_q ≥ N ≥ D_{q+1}となるqが存在するとき、このqはkパラメータの最適値の内の1つである。』

ここでいう最適値とは、Golomb-Rice符号化したときの全体の符号量が最小になるという意味である。また、mはゼロでないB_pの中で最も大きなpの値であり、このmに対応するS_mはゼロある。（一番大きな量子化絶対値をmビット右シフトすると0になるため）よって、D_q = B_q ≤ Nである。

【0048】kパラメータがqからq+1になると符号量がN-D_{q+1}だけ増加し、kパラメータがqからq-1になると符号量がD_{q-1}だけ増加する。但し、D_{q+1} = NまたはD_q = Nの時は符号量は増えず最小のままである。D_{q+1} = Nの時はq+1も最適値となり、D_q = Nの時はq-1も最適値となる。

【0049】最適値は最大3個存在する。その条件はD_q = N-D_{q+1}が成立する時で、言い替えるとS_{q-1} = 0、S_q = N、S_{q+1} = 2Nとなる時である。この時、q-1、q、q+1が最適値となり、全符号量Tは極値（最小値）をとる。よって、kパラメータが上記最適値から遠ざかる程、全符号量は増加する。

【0050】一方、上記qが存在しない時というのは、N > D₁となる時で、この場合kパラメータの値が0で全符号量が最小となる。そして、全符号量はkの値の増加（増加）に対し単調増加となる。

【0051】ここで、前記数列と最適値に関する関係を一般化するために、D₀ = D₁ + Nと定義する。すると、『数列：D₀、D₁、D₂、D₃、⋯、D_{p-1}、D_p、D_{p+1}、⋯、D_qには、D₀ ≥ D₁ ≥ D₂ ≥ D₃ ≥ ⋯ ≥ D_{p-1} ≥ D_p ≥ D_{p+1} ≥ ⋯ ≥ D_qという関係があり、D_q ≥ N ≥ D_{q+1}となるqが存在し、このqはkパラメータの最適値の内の1つである。』ということが言え、最適値qの存在が明確なものとなった。

【0052】これまでの説明から、kパラメータの変化に対する全符号量Tkの局所的な最小値は、kパラメータの全範囲におけるグローバルな最小値でもあることが分る。よって、kパラメータの最適値の探索は、kの値すべてについて全符号量Tkが最小になるかどうかを評価する必要はなく、局所的にTkが最小値となるkの値を見つけ出すだけでよいと言える。これにより、kパラメータの探索方法の簡略化が図れる。

【0053】以上で述べたさまざまな関係から、kパラメータの最適値を決定するいくつかの方法（発明の実施形態）が考えられるが、本第1の実施形態の例を図5のフローチャートに示す。

【0054】まず、ステップS501では、可変長符号部分の符号量 S_p ($p=0, 1, 2, \dots, m-1$)を計算し、ステップS501aでは、ゼロでない S_p のpの最大値に1を加算してmを求める。

【0055】ステップS502では、iに上記mを設定し、ステップS503では、 $D_i = S_{i-1} - S_i$ を計算する。

【0056】ステップS504では、 D_i をNと比較し、 $D_i < N$ なら、ステップS505へ、そうでなければ、ステップS507へ行く。

【0057】ステップS505では、 $i = i - 1$ を計算してkパラメータの候補値を更新する。そして、ステップS506では、更新したiの値が0かどうかを判定し、 $i = 0$ であればステップS507へ行き、 $i = 0$ でなければステップS503へ行ってループ処理を継続する。

【0058】ステップS507では、kパラメータの値をiに決定する。

【0059】ステップS501における、可変長符号部分による符号量 S_p の計算プログラム例（C言語で記述）を図6に示す。この例では、 $V=0$ のときにも最低1回の加算を行なうが、i f文で $V \neq 0$ を検出したときだけ、do〜while文を実行するようにしてもよい。

【0060】サブブロック内の全て（N個）のVに対して図6の処理を行なったら次のステップS501aでkパラメータの上限候補値mを求める。この値は、前記Vの理論上の最大値から図6における変数pの理論上の最大値をあらかじめ計算しておき、 $S[\]$ の値を順に評価することで求められる。

【0061】そして、ステップS502において、該mをiに設定する。

【0062】その次にステップS503にて $D_n = S_{n-1} - S_n$ を計算する。この D_i は全ての D_i の中で一番小さく、最大でもNであり、Nより大きくなることは有り得ない。これは最も大きな値Vの最上位の有意ビット（0で無いビット）のビットプレーンに「1」が存在する個数を表わしている。

【0063】よって、ステップS504で、 $D_{i-(n-1)} <$

Nと判定され、ステップS505へ移る。

【0064】ステップS505でiの値はmから1つ減らされ $m-1$ となり、次のステップS506でiの値が0かどうか判定されるが、ここでは $i \neq 0$ としてステップS503へ戻る。

【0065】上記、ステップS503〜S506の処理を繰り返し行ない、 $D_i > N$ または $i = 0$ になるとkパラメータの値を変更しても符号量が減らないため、ステップS507に抜けて、kパラメータの値をiに決定する。

【0066】以上のようにして、図1における符号化パラメータの選択部で処理を行ない、該パラメータに基づき、既に説明したGolomb Rice符号化処理部106を実行する。符号化されたデータは、次段のビットプレーン走査部107に出力される。

【0067】ビットプレーン走査部107は、量子化値毎に符号化・出力された符号化データを受け取り、それをビットプレーン単位に並び替えて出力するのである。これを、上述した周波数成分（サブブロック）単位で行なう。

【0068】まず、Golomb Rice符号化部で生成された符号化データをkパラメータでの値で決まる固定長符号部分と可変長符号部分に分離する。符号にはその外に1ビット固定の負号ビットがある。

【0069】走査する順序は、重要な情報からで、まず最初に負号ビットをサブブロック内の全量子化値に対して走査する。次に、量子化値の上位ビットの情報を有する可変長符号部分を上位ビットプレーンから順に走査する。最後に固定長符号部分を上位ビットプレーンから順に走査する。各量子化値に対するGolomb Rice符号（但し、可変長符号部分と固定長符号部分とを分離したもの）と走査順序を図7に示す。

【0070】可変長符号部分の走査についてはもう少し詳しい説明を付け加える。可変長符号はその名の通り長さが一定でないため、短い符号と長い符号をどのようにそろえて並べるかという点が一意に決まらないが、ここでは、可変長符号の先頭ビットが同じプレーンになるように並べるものとする。すなわち、可変長符号は図7における可変長符号領域の上から順に上めるようにして並べる。上記ビットプレーン情報を上位ビットから順に走査すると、短い符号では途中のプレーンから情報（符号ビット）が無くなるので情報の無い符号は走査をスキップする。よって、下位のビットプレーンへ行くにつれ、走査する情報（符号ビット）が少なくなる。

【0071】上記順序（負号ビット、可変長符号、固定長符号）で走査したビット情報の並びを図8に示す。

【0072】このビットプレーン走査をサブブロック単位でLL, HL3, LH3, HH3, HL2, LH2, HH2, HL1, LH1, HH1の順に行なう。

【0073】このビットプレーン走査部から出力される

ビット列を次の符号出力部109にて順次転送する。転送相手は、ハードディスクやDVD等の記録メディアであっても良いし、インターネット、一般公衆回線、無線回線等のインターフェースであっても良い。

【0074】なお、上記実施の形態において生成された符号化データには、後写時に必要となる各種情報、すなわち、画像サイズ、1画素あたりのビット数、各周波数成分に対する量子化ステップ、kパラメータの値等を付属情報として符号化データに適宜付加する。

【0075】〔第2の実施形態〕上記第1の実施の形態における符号化パラメータの選択方法、すなわち、kパラメータの生成方法とは若干異なる方法を図9に示し説明する。なお同じ役割を担う処理ステップには図5のものと同一の番号を付加している。

【0076】本第2の実施形態は、前記(6)式： $D_p = S_p + B_p$ を利用する。よって、ステップS501の替わりにステップS901、ステップS503の替わりにステップS903にて処理を行なう。

【0077】ステップS901では、可変長符号部による符号量 S_p ($p=0, 1, 2, \dots, m-1$)と各ビットプレーンにおける「1」の数 B_p ($p=1, 2, \dots, m$)を計算し、ステップS903では、 $D_1 = S_1 + B_1$ を計算する。

【0078】ステップS901の S_p と B_p の計算プログラム例(C言語で記述)を図10に示す。該プログラムにおけるdo〜whileループ文の第1遡目では S_{p-1} 、 B_{p-1} に対して累積加算処理を行ない、第p遡目では S_p 、 B_p に対して累積加算処理を行なうようにしている。

【0079】本第2の実施形態の最大の特徴は、 D_1 を前記(6)式で計算することにある。そのために必要な S_1 と B_1 をステップS901にてあらかじめ求めておき、ステップS903にて D_1 を計算する。その他の処理は前記第1の実施形態と同じであるため、説明は省略する。

【0080】〔第3の実施形態〕本第3の実施形態は、前記(5)式： $S_p = 2 \times S_{p-1} + B_{p+1}$ の関係を利用する。すなわち、前記第2の実施形態におけるステップS901の処理で、 S_1 と B_2 ($p=0, 1, 2, \dots, m$)を計算する部分を、以下に示すステップS911〜ステップS913の処理に置き換える。

【0081】ステップS911では、 B_p ($p=0, 1, 2, \dots, m$)を計算する。

【0082】ステップS912では、ゼロでない B_p のpの最大値mを求める。

【0083】ステップS913では、 $S_m=0$ に初期化し、 $S_{p-1}=2 \times S_p + B_p$ ($p=m, m-1, m-2, \dots, 1$)を計算する。

【0084】その他の処理は、前記第2の実施形態と同じであるので、説明を省略する。なお、ステップS913の処理を「 $S_{m-1}=B_m$ に初期化し、 $S_{p-1}=2 \times S_p$

+ B_p ($p=m-1, m-2, \dots, 1$)を計算する。」と変更してもよい。

【0085】〔第4の実施形態〕本第4の実施の形態は、前記(6)式の関係： $D_p = S_p + B_p$ から容易に導き出せる $D_p \leq S_p$ という関係を利用する。 $D_p \leq S_p$ という関係より、 $S_p < N$ であれば、 $D_p < N$ であると言える。

【0086】そこで、図11に示すフローチャートのよ様に、 $S_1 < N$ の関係が成り立つかどうかを調べるステップS1101を前記第2の実施の形態を示す図9のステップS903の前に設け、該関係が成り立てば、ステップS903とステップS504をパスするようにした。

【0087】なお、本実施形態は、前記第1の実施形態に適用することも可能である。その場合、ステップS1101を図5のステップS503の前に設け、該関係が成り立てば、ステップS503とステップS504をパスするようにする。

【0088】〔第5の実施形態〕上記第1、第2の実施の形態における符号化パラメータの選択方法とは異なる、更に別の方法について説明する。

【0089】本実施の形態では、kパラメータの値が2つ異なる S_{p-2} と S_p 間の差、すなわち、 $DD_p = S_{p-2} - S_p$ を利用する。

【0090】既に説明したように、 DD_p が2Nより小さくなら、kパラメータの値をpからp-2に変えることで可変長符号量が $2N - DD_p$ だけ減るので、kパラメータの値はpよりp-2にした方がよい。逆に、 $DD_p \geq 2N$ ならkパラメータの値はpの方がよいと言える。

【0091】よって、該 DD_p を用いば、kパラメータの候補値を2単位で高速に更新できる。本実施形態の処理方法を示すフローチャートを図12に示す。同じ役割を担う処理ステップには図5のものと同一の番号を付加している。

【0092】本実施形態では、前記第1の実施形態における図5のステップS502とステップS503の間に別のループ処理を行なうステップS1201〜ステップS1204を設けている。

【0093】ステップS1201では、本実施形態の一番の特徴である $DD_i = S_{i-2} - S_i$ を計算する。ステップS1202では、 DD_i と2Nとを比較する。ステップS1203では、iを2単位で更新する。ステップS1204では、iが1以下であるかどうかを判定する。ステップS1201〜ステップS1204によるループ処理は、前記第1の実施形態におけるステップS503〜ステップS506によるループ処理と大変よく似ている。

【0094】違いはiを更新する単位が2であるということである。ステップS1201で計算した DD_i をス

ステップS1202でNと比較することにより、iを2つ更新してもよいかどうか判定し、更新可能であれば、ステップS1203にて $i = i - 2$ によりiの値を更新する。

【0095】 $i \leq 1$ ではステップS1201を実行できないので、(何故なら S_i が存在しないため)ステップS1204にて、 $i \leq 1$ を検出したら次のステージの処理(ステップS503)へ移り、 $i > 1$ であれば、ステップS1201へ戻ってループ処理を継続する。

【0096】上記ループ処理を抜けてきた時というのは、kパラメータをiから $i - 2$ にしても符号量が減らないという状態である。しかし、kパラメータをiから $i - 1$ にすると符号量が減る可能性がある。

【0097】そこで、kパラメータをiから $i - 1$ に変更可能かどうかを調べるため、ステップS503で $D_i = S_{i-1} - S_i$ を計算し、ステップS504で D_i をNと比較し、 $D_i < N$ なら、iから $i - 1$ に変更できるので、ステップS505に移り、そうでなければ、それまでのiをそのまま引き継いでステップS507へ行き該iの値をkパラメータの値とする。

【0098】本実施形態では、まず最初にiを2単位で更新したが、 $S_{i-4} - S_i$ を用いれば、4単位で更新することもできるし、他の計算式を用いれば別の単位で更新することも可能である。

【0099】4単位で更新した場合、最初のループ処理を抜けた時点で、最適なkパラメータの可能性としては、 $i - i - 3$ の4つが考えられる。そこで、次のステージで、この4つの値のどれかに特定するため、1単位で更新可能かどうかをループ処理で最大3回判定し、iの値を更新する、といった処理方法も考えられる。

【0100】[第6の実施形態] 符号化パラメータの選択方法として、更に異なる方法を図13に示し説明する。なお同じ役割を担う処理ステップには図5のものと同じの番号を付加している。

【0101】本実施形態は、 D_i を使用せずに、 $SN_p = S_p + N$ ($p = 0, 1, 2, 3, \dots, m$) という値を用いることが特徴である。

【0102】ステップS1301では、上記 SN_p を計算し、ステップS1302では、 S_{i-1} と SN_i を比較する。

【0103】その他の処理は前記第1の実施の形態と同じである。

【0104】ステップS501で計算した S_p ($p = 0, 1, 2, 3, \dots, m$) の値に対して、Nを加算した値 SN_p をステップS1301で計算した後、ステップS502でiの初期化を行なう。

【0105】その後、kパラメータの候補値であるiを $i - 1$ へ更新可能かどうかを調べるため、ステップS1302にて、 S_{i-1} と SN_i を比較する。 $S_{i-1} < SN_i$ であれば符号量が減るので、iを $i - 1$ へ更新するためス

テップS506へ行き、そうでなければ、ステップS504に抜けkパラメータの値をiに確定する。

【0106】iを $i - 1$ へ変更した後、ステップS506にて、 $i = 0$ かどうか判定する。 $i = 0$ になるとiの値をそれ以上更新できないので、ループ処理を抜けステップS504へ行く。そうでなければ、ステップS1302へ戻り、iの更新処理を継続する。

【0107】[第7の実施形態] これまで説明した実施形態は、すべてループ制御変数iの初期値をmとしてきたが、本実施の形態では、該制御変数iの初期値を0に設定するものである。

【0108】基本的には、これまで述べてきた全ての実施形態に適用可能であるが、特に第1の実施形態である図5を基にして、本実施の形態を説明する。iの初期値を0とする本実施形態を図14に示す。

【0109】同図において、ステップS1402では、iに初期値0を設定し、ステップS1404では、 D_i をNと比較し、 $D_i > N$ ならステップS1405へ、そうでなければステップS1407へ行く。ステップS1405では、iに1を加算し、ステップS503に戻る。

【0110】ステップS1407では、kパラメータの値を $i - 1$ に決定する。

【0111】本実施形態では、これまでの実施形態が必要であった、mを求めるためのステップS501a、iが0かどうかを判定するステップS506が必要なくなった。

【0112】以前述べたように $D_p \leq N$ であるため、iの値がmで大きくなるか、それ以前にある、 $D_i \leq N$ という関係になるため、ステップS1404の分岐先は最後は必ずステップS1407になり、ここでkの値が確定する。

【0113】ステップS1407でkパラメータの値を、iではなく $i - 1$ に設定する理由は、ステップS1404からステップS1407に分岐した時点での諸関係が、 $D_{i-1} > N \geq D_i$ であるからである。前述したように、該関係でのkパラメータの最適値は $i - 1$ である。

【0114】本実施形態は、これまでの中で最もシンプルな処理形態をとることができる。

【0115】[第8の実施形態] 本実施形態は、前記制御変数iの初期値を、それ以前に行なう比較処理の結果によって変えるというものである。

【0116】kパラメータの最適値は、 $i = m$ に近いところに存在する場合もあれば、 $i = 0$ に近いところに存在する場合もある。前者であれば $i = m$ から、後者であれば $i = 0$ から探索するのが合理的というものである。

【0117】ループ処理に入る前にまずその判定をしなければならない。 $h = f \text{ floor } \{m/2\}$ に対する、 D_h を計算し、該 D_h とNとを比較する。 $D_h < N$ であれば、iを0に設定して処理を行ない、そうでなければ、

i を m に設定して処理を行なう。というのが本実施形態の概略である。

【0118】本実施形態を図15に示し、もう少し詳しく説明する。同図において、ステップS1502では、 $h = \text{floor} \{ m/2 \}$ を計算する。ステップS1503では、上記hに対応する D_h を計算する。

【0119】ステップS1504では、 D_h とNとを比較し、 $D_h < N$ であればステップS1402へ分岐し、そうでなければ、ステップS502へ分岐する。

【0120】その他の処理ステップは図5あるいは図14の同一番号処理ステップと同じである。

【0121】ステップS1402以降の処理は、前記第7の実施形態とまったく同じ処理であるが、ステップS502以降の処理は、基本的には前記第1の実施形態と同じであるが、若干違いがある。その違いは、ループ処理において、i が所定の値に達したかどうかの判定（ステップS506に相当する）が無いことである。

【0122】何故判定が無いかというと、該ループ処理においてiの値が初期値mから1ずつ減少して上記hに達すると、ステップS504の判定で該ループ処理を抜けるステップS507へ確実に分岐することが分かっているからである。これはステップS1504からステップS502へ分岐したときの条件 ($D_h \geq N$) から、明白である。

【0123】同様に、ステップS503、S1404、S1405によるループ処理を抜ける時のiの値の上限はhである。これはステップS1504からステップS1402へ分岐したときの条件 ($D_h < N$) から分かる。

【0124】本実施形態では、kパラメータを決定するまでのループ処理の上限ループ回数を約半分にすることができるので、より高速にkパラメータを決定することができる。

【0125】また、本実施形態の応用として、kパラメータを決定するループ処理への分岐数を増やせば、各ループ処理の上限ループ回数を更に減らすことができるので、より高速にkパラメータを決定することができる。

【0126】〔第9の実施形態〕前記第9の実施形態のような初期値の設定方法では、制御変数iが増加する場合のループ処理と減少する場合のループ処理の2つを用意しなければならない。そこで、該2つのループ処理を共通化した場合の処理形態を図16を用いて説明する。

【0127】本実施形態では、ステップS1601にて、iの初期値をhに設定する。ステップS1602では、iがhに等しいかどうか判定する。ステップS1603では、iの初期値を0に再設定すると共に、hを-1に設定する。

【0128】その他の処理ステップは前記第8の実施形態：図15の同一番号処理ステップと同じである。

【0129】前記第8の実施形態と同様の演算でhを計

算し（ステップS1502）、ステップS1601にて、iに該hを設定する。次に、ステップS503に移り、 $D_h = S_{h-1} - S_h$ を計算する。そして、ステップS1404にて、 D_h がNより大きいかどうか判定する。ここまでの処理は、フローチャートの構造は違ってはいれるが、前記第8の実施形態と同じである。

【0130】 $D_h > N$ であれば、 $i = h$ を初期値として、制御変数iが増加する場合のループ処理と同様の処理をステップS503、S1404、S1405にて行なう。該ループ処理を抜ける時のiの値は、少なくとも1回はステップS1405 ($i = i + 1$)を計算する)を通過するので、前記hより大きく、ステップS1602の判定を通過して、ステップS1407にて、kパラメータの値をi-1に決定する。

【0131】 $D_h > N$ でない場合は、初回からステップS1602へ分岐する。この時 $i = h$ なので、ステップS1603へ分岐し、iの初期値を0に再設定すると共に、hを-1に設定する。hを-1に設定する理由は1が取り得ない値を設定することで、2度目のステップS1602における判定にひっかからない ($i = h$ とならない) ようにするためである。

【0132】iの初期値を0に再設定した後の処理は、前記第8の実施形態におけるステップS503、S1404、S1405によるループ処理と完全に同じで、該ループ処理を抜ける時のiの値は最大で $\text{floor} \{ m/2 \}$ である。

【0133】変数hの値はステップS1603で-1に設定されているので、ステップS1602にて $i = h$ が成り立つことが絶対にならないため、ステップS1407へ分岐し、kパラメータの値はi-1に決定される。

【0134】前記第8の実施形態に対して、処理を共通化したことによるオーバーヘッドは、iの初期値を0に再設定する際に必要なステップS1602の判定処理が初期化時に1/2回（初期化時にこの判定処理を行なう確率は約1/2であるため）と前記ループ処理を抜ける際に1回（この時は該判定処理を越えて通ることができないため）あるが、処理ルーチンのコンパクトさのメリットの方が大きいと思われる。

【0135】なお、上記各実施形態におけるその構成は図1に従うものとしたが、例えば同図の処理をプログラムによっても実現できる。画像を入力するデバイス、ネットワーク、ビデオカメラ、イメージキャッチャ、ストレージデバイス（フロッピー等）で良いわけであるから、本発明は、例えばこれらのいずれかを有するシステムや汎用情報処理装置（パーソナルコンピュータ等）にも適用できる。

【0136】従って、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体（または記録媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（または

CPUやMPU)が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム(OS)などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0137】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0138】以上説明したように本実施形態によれば、あるパラメータに基づいてシンボルを可変長符号部分と固定長符号部分とで符号化する際、そのパラメータの各々の値に対する可変長符号部分の符号量間の差分値のみを評価して、最適なパラメータを高速に決定できるようになった。

【0139】

【発明の効果】以上説明したように本発明によれば、多値画像を簡易かつ高速に可変長符号化することのできるようになる。

【図面の簡単な説明】

【図1】実施形態の装置のブロック構成図である。

【図2】離散ウェーブレット変換を施す様子を示す図である。

【図3】周波数成分(サブブロック)に用いる量子化ステップを示す図である。

【図4】Golomb-Rice符号化をして得られる符号の一例を示す図である。

【図5】第1の実施形態である処理内容を示すフローチャートである。

【図6】kパラメータの各値に対する可変長符号の符号量Spを計算するプログラムの一例を示す図である。

【図7】Golomb-Rice符号化した符号をビットプレーンで表現した図である。

【図8】最終的に出力されるビットストリームの様子を示す図である。

【図9】第2の実施形態である処理内容を示すフローチャートである。

【図10】符号量Spと各ビットプレーン中の1の数Bpの両方を計算するプログラムを示す図である。

【図11】第4の実施形態である処理内容を示すフローチャートである。

【図12】第5の実施形態である処理内容を示すフローチャートである。

【図13】第6の実施形態である処理内容を示すフローチャートである。

【図14】第7の実施形態である処理内容を示すフローチャートである。

【図15】第8の実施形態である処理内容を示すフローチャートである。

【図16】第9の実施形態である処理内容を示すフローチャートである。

【符号の説明】

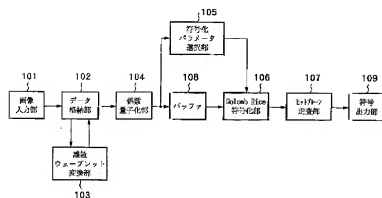
- 101 画像入力部
- 102 データ格納部
- 103 離散ウェーブレット変換部
- 104 係数量子化部
- 105 符号化パラメータ選択部
- 106 Golomb-Rice符号化部
- 107 ビットプレーン走査部
- 108 バッファ
- 109 符号出力部

【図6】

/* サブブロック内のすべての絶対値化した量子化値Vに対して
以下の処理を行なう */

```
int t=V;
int p=0; /* 初期化 */
do{
    S[p+t]+=t; /* Sp=Sp+(V>>p), p=p+1 */
}while(t>>1) /* (V>>p)>0なら処理を継続 */
```

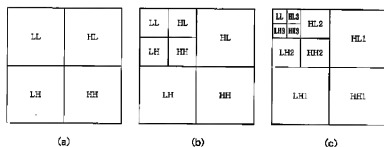
【図1】



【図3】

周波数成分	量子化ステップq
LL	1
HL3	2
LH3	2
HH3	2
HL2	4
LH2	4
HH2	4
HL1	8
LH1	8
HH1	8

【図2】



【図4】

X \ Y	0	1	2
0	1	10	100
1	01	11	101
2	001	010	110
3	0001	011	111
4	00001	0010	0100
5	000001	0011	0101
6	0000001	00010	0110
7	00000001	00011	0111

【図7】

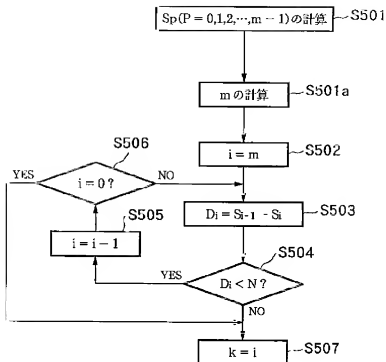
	3	8	-2	-5	4	0	1
符号ビット	0	0	1	1	1	0	0
変換符号	1	0	1	0	0	1	1
変換	0	1	1	1	1	1	1
変換	1	1	1	1	1	1	1
変換	1	0	1	0	0	0	0
変換	1	0	0	1	0	0	1

k=2

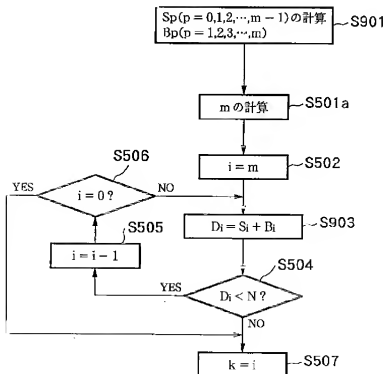
【図8】



【図5】



【図9】



【図10】

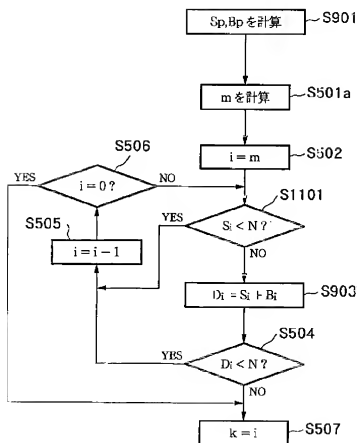
/* サブブロック内のすべての絶対値化した量子化値Vに対して
以下の処理を行なう */

```

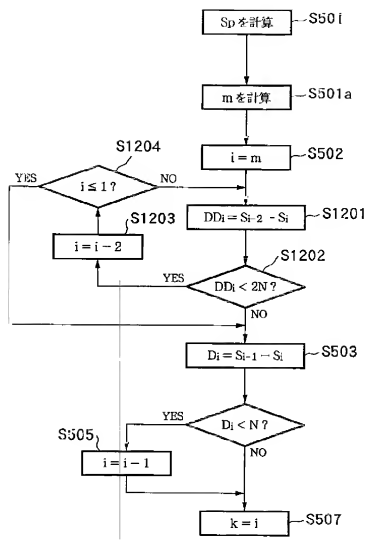
int  t = V ;
int  p = 0 ;          /* 初期化 */
do{
    S[p++] += t ;      /* Sp = Sp + (V >> p), p = p + 1 */
    B[p] += (t & 1);    /* Bp = Bp + ((V >> p) & 1) */
}while(t >>= 1)        /* (V >> p) > 0 なら処理を継続 */

```

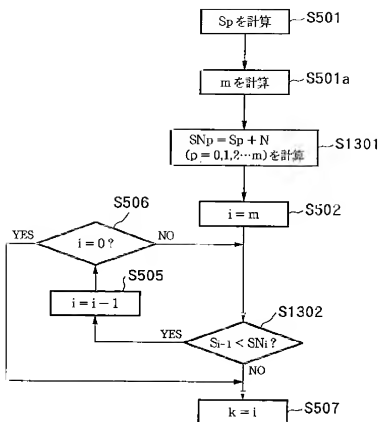
【図11】



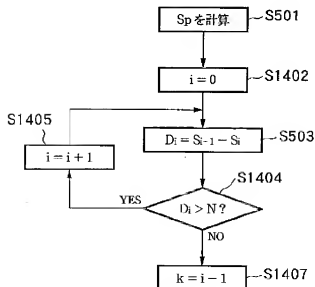
【図12】



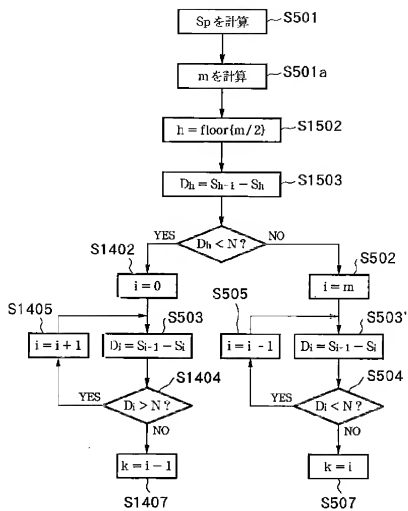
【図13】



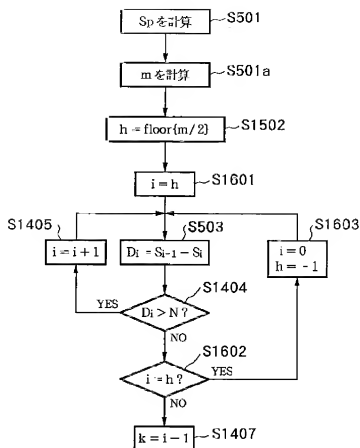
【図14】



【図15】



【図16】



フロントページの続き

Fターム(参考) 5C059 KK11 MA24 MA35 NC15 ME01
 ME13 ME17 PP15 PP16 SS13
 SS20 SS26 TA48 TB08 TC00
 TD05 UA02 UA34
 5C078 AA09 BA21 BA53 CA31 CA35
 DA01 DA07 DB07 DB19
 5J064 AA03 BA09 BA16 BB01 BC02
 BC16 BC25 BD01
 9A001 EE04 GG17 HH27 JJ71 KK54